CODE CONVERSION

by Donald Femling

INTRODUCTION

To obtain a clear understanding of code conversion, it is first necessary to define a "natural" binary decimal code (BCD). BCD coding takes each digit of a decimal number and codes it separately into binary. This implies four bits for each decimal digit. Since four binary bits can represent sixteen different quantities and are being used to represent zero through nine, there is considerable redundancy built-in into this code. As compared with binary representation, storage capacity is wasted in BCD, and the hardware required is usually more costly to implement. Despite these disadvantages, many systems have been designed to use some form of this code, because of the ease of display.

If a machine has to interface with several different input devices, and when using as many as three or four different codes, it becomes necessary to convert the inputs to a code the machine can handle. The code selected is usually determined either by how costly it is to implement the operations required by the machine and the interface converters by the speed required, or by the storage capacity available.

There are innumerable four-bit codes which can be set up to represent decimal digits; $16!/6! = 29,059,430,400$, to be exact. This article, however, will deal primarily with weighted codes, since they are the most commonly used. These codes are of the form $D = d_4W_4 + d_3W_3 + d_2W_2 + d_1W_1$, where d is a binary digit (either 1 or 0), and W is an integer weight assigned to that digit position.

MAN-MACHINE INTERFACES

Categorized with code converters are encoders and decoders. Encoders are circuits or transducers that convert decimal data to binary coded decimal form, and imply a conversion process prior to some binary operation. Decoders convert binary coded data to decimal form, and imply conversion after the binary operation has been completed.

Encoders

Figures 1 through 3 show a method of converting single-line decimal data (i. e., each bit represented by a single wire,

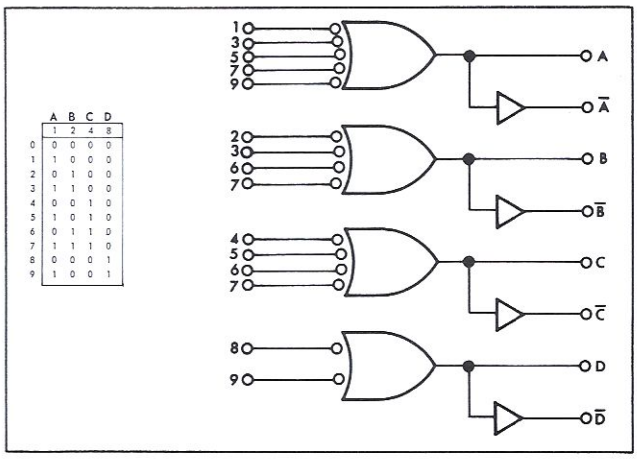

Fig. 1. Decimal-to-binary (1248) weighted conversion.

0 through 9) into a BCD code. Two of the most common binary-to-decimal codes were used, as may be noted by the truth table associated with each circuit. The truth table of Fig. 2 will show that it is the self-complementing code, whereas Fig. 3 is a non-complementing 1242* code.

COPYRIGHT, FAIRCHILD SEMICONDUCTOR, 1967 • PRINTED IN U.S.A. 2320-274-27 35M

313 FAIRCHILD DRIVE, MOUNTAIN VIEW, CALIFORNIA, (415) 962-5011, TWX: 910-379-6435

FAIRCHILD SEMICONDUCTOR
A DIVISION OF FAIRCHILD CAMERA AND INSTRUMENT CORPORATION

Fairchild cannot assume responsibility for use of any circuitry described other than circuitry entirely embodied in a Fairchild product. No other circuit patent licenses are implied.

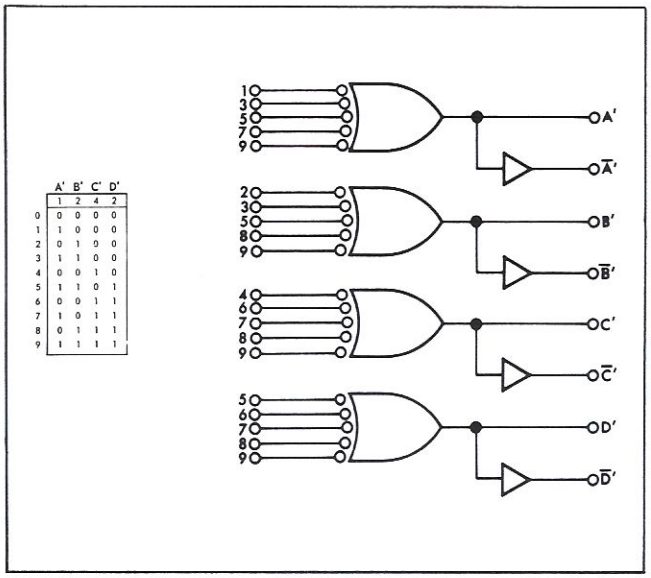

Fig. 2. Decimal-to-binary (1242*) self-complementing conversion.

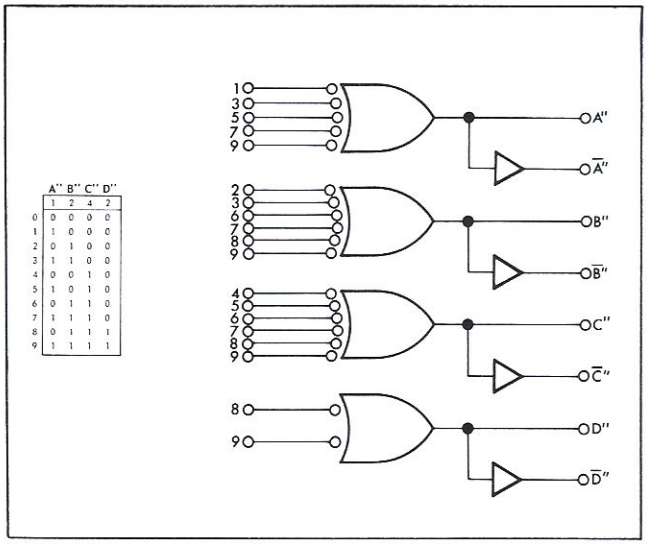

Fig. 3. Decimal-to-binary (1242*) noncomplementing conversion.

Decoders

Three of the most obvious decoders will be the binary to decimal converters shown in Fig. 4 through Fig. 6. Some of the less obvious ones are a binary to quibinary and a binary to seven-line converter. Both codes are used for luminescent numerical displays. These displays accept a given code and convert it within their structure to decimal information for visual readout. The seven-line code (Fig. 7) is used in a variety of seven-segment displays, and the quibinary code (Fig. 8) is used in some gas-discharge tubes. The conversion from quibinary to binary (Fig. 9) is more simple than is the reverse process, due to the larger number of "don't care" states.

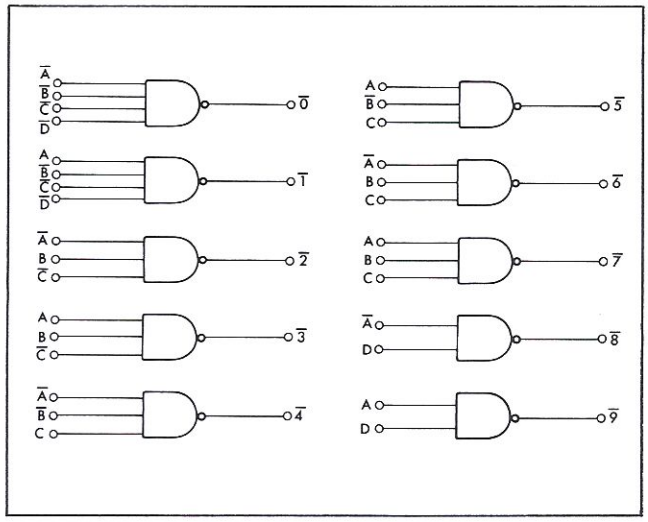

Fig. 4. Binary (1248)-to-decimal conversion.

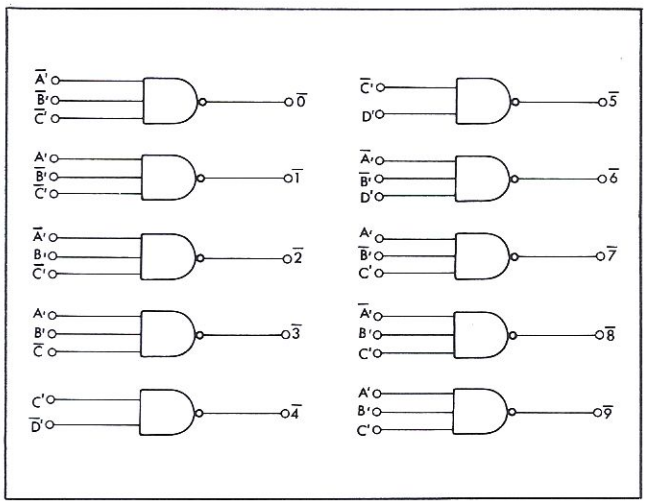

Fig. 5. Binary (1242* complementing)-to-decimal conversion.

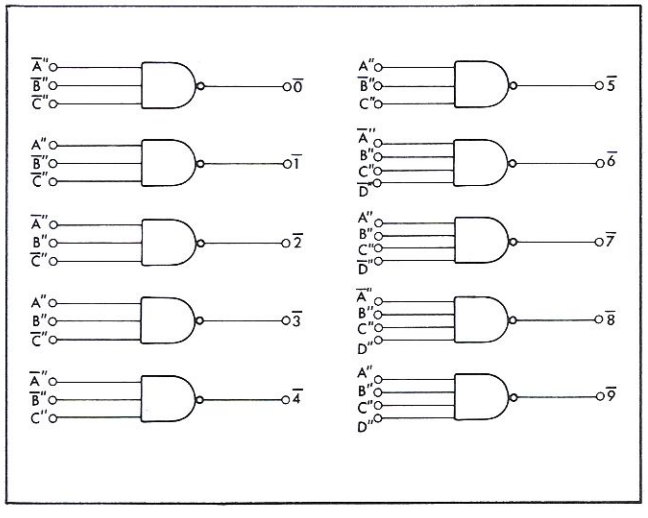

Fig. 6. Binary (1242* noncomplementing)-to-decimal conversion.

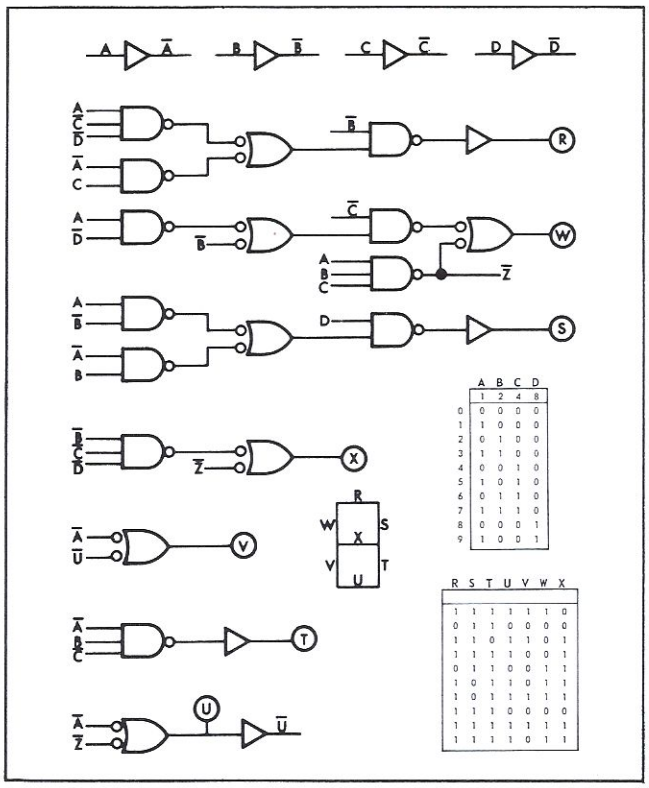

Fig. 7. BCD to seven-bit conversion.

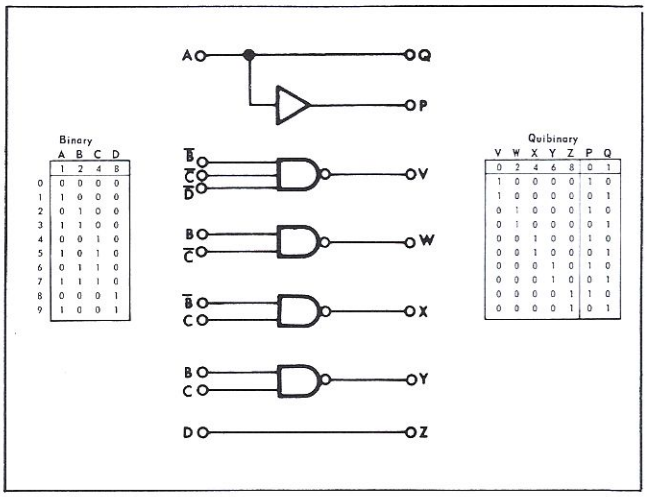

Fig. 8. Binary-to-quibinary conversion.

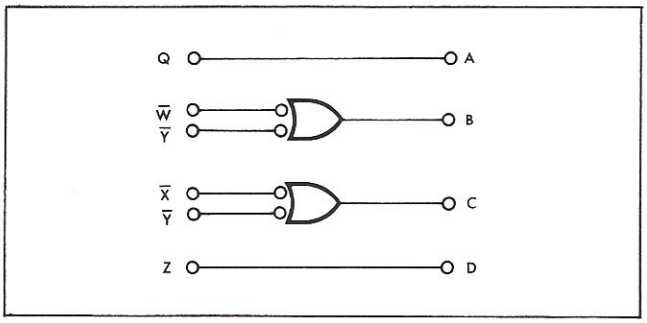

Fig. 9. Quibinary to binary-coded decimal.

MACHINE-TO-MACHINE INTERFACES

Within any given system, (Fig. 10, 11, 12, 13) several codes can exist. One code could be used in performing arithmetic operations. Another code could be used in the transmission of data. And still another may be used in the conversion of analog to digital information. The code will vary depending upon the ease with which an operation may be performed using the particular code. This, in turn, affects the cost

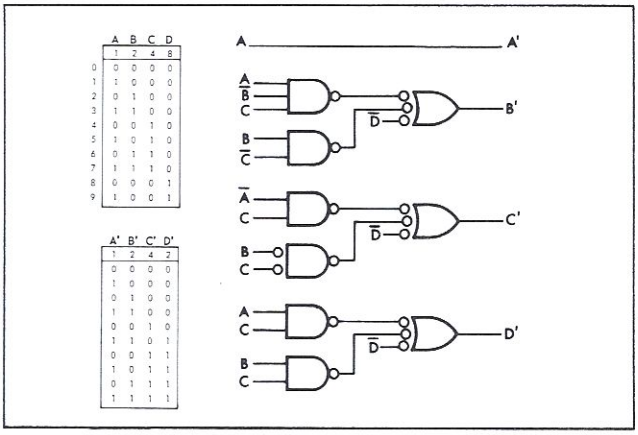

Fig. 10. Binary (1248) to binary (1242*) self-complementing.

of a given task. In every large system, especially if a great deal of peripheral equipment is attached, or when it is interfaced with another system, code conversion is usually essential. Three of the most common machine codes are binary

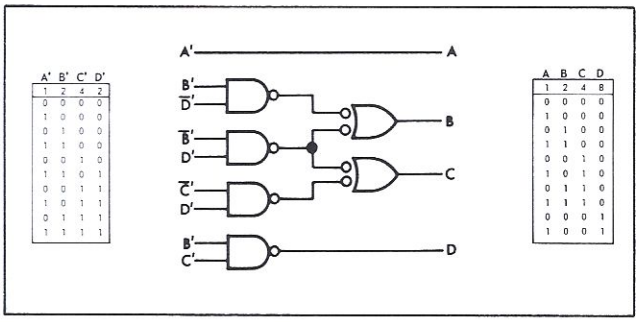

Fig. 11. Binary (1242*) self-complementing to binary (1248) conversion.

1248 weighted, binary 1242* weighted, and biquinary. Binary 1248 is the classic binary weighting system, found in most computers. Binary 1242* is a binary weighted code that becomes valuable because only three different weights are used. Therefore, when buying the components required to represent these weights, only three values need be purchased, resulting in considerable cost saving in a large system. This code may also be self-complementing, which can be a valuable characteristic in some logic operations. The self-complementing code is usually found in equipment that performs a large amount of analog to digital conversion, such as in mea-

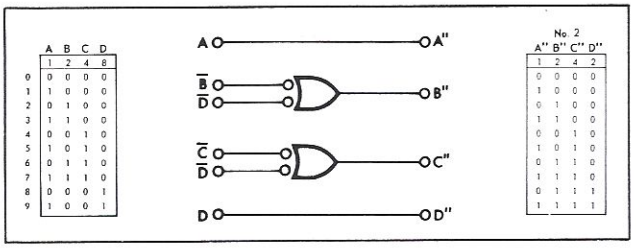

Fig. 12. Binary (1248) to binary (1242*) noncomplementing conversion.

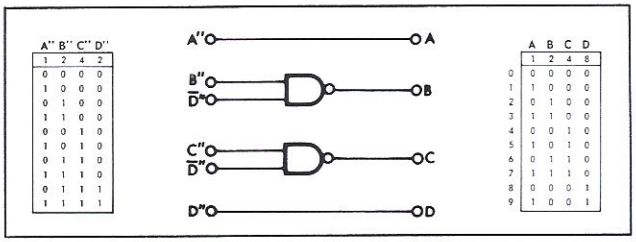

Fig. 13. Binary (1242*-2) noncomplementing-to-binary (1248) conversion.

suring and recording instruments. The biquinary code will usually be found in computers and various calculating machines. It makes possible some simplifications in circuitry designed to perform arithmetic operations. Another reason for using such a seven-bit code is the ease of detecting errors. From an examination of the truth table, (Fig. 14 and 15) it may be noted that each digit is represented by only two bits. One of these is always located in the first two bits and the other is always in the last five bits. This arrangement, in

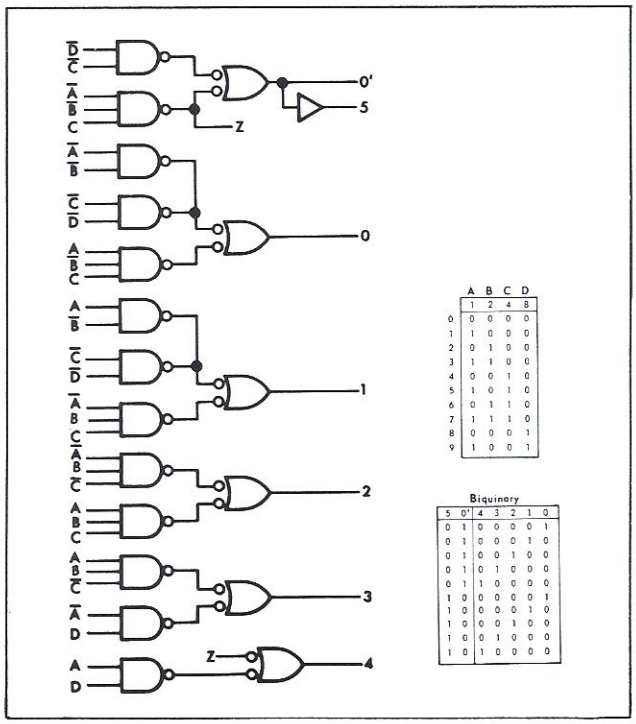

Fig. 14. Binary (1248) to biquinary logic diagram.

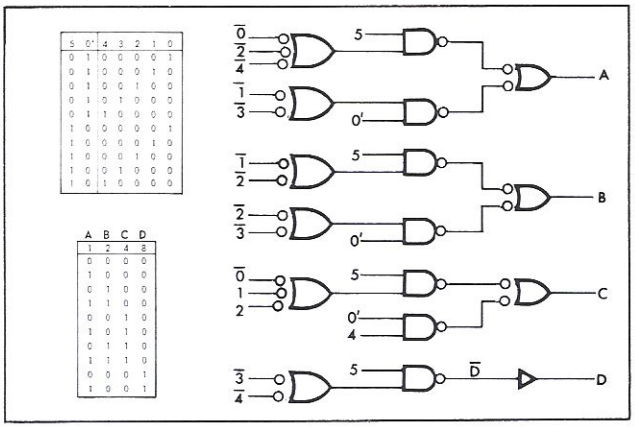

Fig. 15. Biquinary-to-binary (1248) conversion.

addition to making errors easier to detect, also allows fairly simple decoding. A two-input AND gate is the only requirement to decode any digit.

Another way of accomplishing biquinary to binary conversion is to decode the biquinary into its decimal form and then to convert the decimal data into its binary form (Fig. 16). Either method requires about the same amount of hardware.

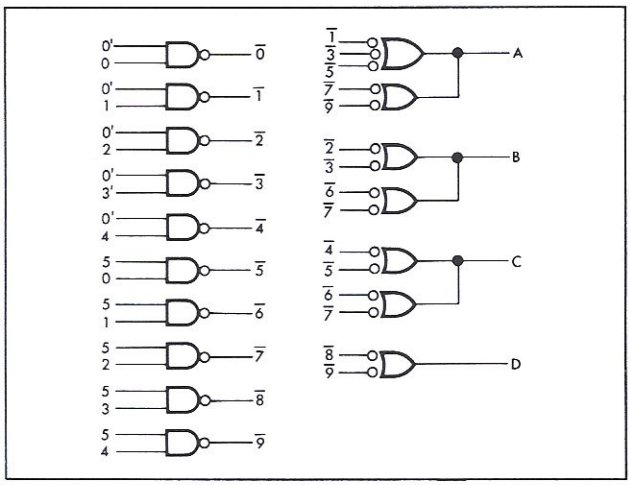

Fig. 16. Biquinary-to-decinal-to-binary conversion.

The exact hardware (can count) required for each circuit will depend on the integrated circuit family selected. These circuits may be implemented with any digital integrated circuit family. Every attempt has been made to show these circuits as simple collections of gates; however, in some cases the gate grouping has been selected to fit standard configurations that exist in DTμL or RTμL families. There is a large number of possible decimal codes; those used in this bulletin are representative of the most useful encountered in the more simple systems using four-bit codes. The circuits designed will be used when two or more of these codes are encountered within a system, or between two systems that must exchange digital data but which have different codes.